ROBOT LOCALIZATION: AN INTRODUCTION

1. INTRODUCTION

Robot localization provides an answer to the question: *Where is the robot now?* A reliable solution to this question is required for performing useful tasks, as the knowledge of current location is essential for deciding what to do next (1,2). This article focuses on the solutions to the robot localization problem when the map of its environment is available. The problem then becomes one of estimating the robot pose (position and orientation) relative to the coordinate frame in which the map is defined. Typically, the information available for computing the robot location is gathered using onboard sensors, while the robot uses these sensors to observe its environment and its own motion. Given the space limitations, alternative scenarios where sensors such as surveillance cameras are placed in the environment to observe the robot or the robot is equipped with a receiver that provides an estimate of its location based on information from an external source (e.g., a Global Positioning System (GPS) that uses satellites orbiting the earth) are excluded from the following discussion.

A mobile robot equipped with sensors to monitor its own motion (e.g., wheel encoders and inertial sensors) can compute an estimate of its location relative to where it started if a mathematical model of the motion is available. This is known as odometry or dead reckoning. The errors present in the sensor measurements and the motion model make robot location estimates obtained from dead reckoning more and more unreliable as the robot navigates in its environment. Errors in dead reckoning estimates can be corrected when the robot can observe its environment using sensors and is able to correlate the information gathered by these sensors with the information contained in a map. How this can be achieved within a probabilistic framework will be discussed in this article.

The formulation of the robot localization problem depends on the type of the map available as well as on the characteristics of the sensors used to observe its environment. In one possible formulation, the map contains locations of some prominent landmarks or features present in the environment and the robot is able to measure the range and/or bearing to these features relative to the robot. Alternatively, the map could be in the form of an occupancy grid that provides the occupied and free regions of an environment and the sensors on board the robot measures the distance to the nearest occupied region in a given direction. As the information from sensors is usually corrupted by noise, it is necessary to estimate not only the robot location but also the measure of the uncertainty associated with the location estimate. Knowledge of the reliability of the location estimate plays an important role in the decision-making processes used in mobile robots as catastrophic consequences may follow if decisions are made assuming that the location estimates are perfect when they are uncertain. Bayesian filtering (3) is a powerful technique that could be applied to obtain an estimate of the robot location and the associated uncertainty. Both extended Kalman filter (EKF) and particle filter provide tractable approximations to Bayesian filtering and they are the focus of this article.

The remainder of this article is structured as follows. Section 2 provides the mathematical models for describing the robot motion and the relationships between the sensor measurements and the robot location for both feature-based and occupancy grip-based maps. Section 3 presents an algorithm based on the EKF for robot localization using a feature map. Section 4 presents a particle filter for locating a robot in a grid map. Section 5 presents a brief discussion of alternative localization techniques that have been proposed in the robotics literature. The mathematical background to estimation theory and two alternative robot localization techniques are presented in the appendices. The MATLAB code of the localization algorithms for the simple examples are available at https://github.com/UTS-CAS/Robot-Localization-examples and also at http://onlinelibrary.wiley.com/doi/10.1002/047134608X.W8318/Robot-Localization-examples.

2. VEHICLE MODEL AND SENSOR MODELS

The mathematical ~ models describing the behavior of the robot and the sensors mounted on it are the most important components in the formulation of the robot localization problem. The vehicle kinematic model describes the equations governing the robot motion in response to control actions. Figure 1 illustrates a differential drive robot operating on a two-dimensional plane where the forward velocity and the angular velocity of the robot body can be controlled using two motors that drive the two wheels. The differential equation that describes how the robot position and orientation evolve with time as a function of its forward and angular velocity is known as the robot motion model.

The relationship between the observations from the sensors and the location of the robot in the map is known as the sensor model. The sensor model is dependent on the characteristics of the sensor mounted on the robot as well as on the way the map of the environment is represented. As discussed in Section 1, the map of the environment is typically defined either using coordinates of known landmarks or features, or in the form of an occupancy grid where the status of each grid cell defines whether the area represented by the cell is occupied or free space. Figure 2 illustrates a map with four landmarks, while Figure 3 shows an occupancy grid map where the occupied areas are shaded.

In the following sections, a vehicle model and the sensor models typically used for robot localization are described. For simplicity and clarity of notation, it is assumed that the robot has three degrees of freedom and is moving in a two-dimensional plane.

J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*.
DOI: 10.1002/047134608X.W8318

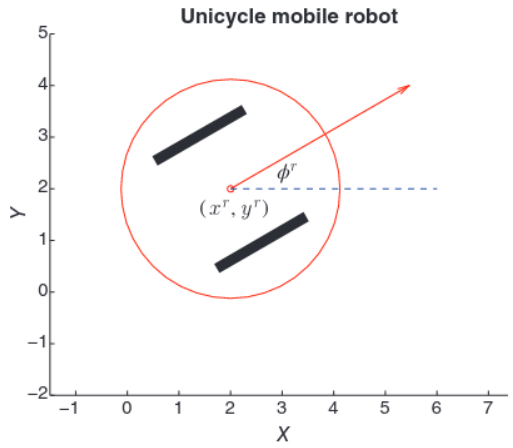

Figure 1. A differential drive robot operating in a two dimensional plane.

2.1. Vehicle Model

The kinematic equations governing the motion of the differential drive robot illustrated in Figure 1 are given by

$$\begin{aligned} \dot{x}^r(t) &= (v(t) + \delta v(t))\cos(\phi^r(t)) \\ \dot{y}^r(t) &= (v(t) + \delta v(t))\sin(\phi^r(t)) \\ \dot{\phi}^r(t) &= \omega(t) + \delta\omega(t) \end{aligned} \tag{1}$$

where the coordinates $x^r(t)$ and $y^r(t)$ describe the position of the center of the mobile robot at time t, the orientation $\phi^r(t)$ is the angle between the heading of the robot and the x-axis of the fixed global coordinate frame. $\dot{x}^r(t)$ denotes the derivative of $x^r(t)$ with respect to time t. The forward velocity $v(t)$ and angular velocity $\omega(t)$ are the control inputs of the robot. $\delta v(t)$ and $\delta\omega(t)$ are the differences between the intended control value and the actual control values (control noises) and are assumed to be zero-mean Gaussian.

Discretizing the continuous-time motion model equation 1 with a sampling time ΔT and the Euler method results in

$$\begin{aligned} x^r_{k+1} &= x^r_k + (v_k + \delta v_k)\Delta T\cos(\phi^r_k) \\ y^r_{k+1} &= y^r_k + (v_k + \delta v_k)\Delta T\sin(\phi^r_k) \\ \phi^r_{k+1} &= \phi^r_k + (\omega_k + \delta\omega_k)\Delta T \end{aligned} \tag{2}$$

where (x^r_k, y^r_k, ϕ^r_k) is the robot location at time step k, v_k is the velocity at time k, and ω_k is the angular velocity at time k, and δv_k and $\delta\omega_k$ are the discrete time velocity noises and angular velocity noises, respectively.

2.2. Sensor Model for Landmark-Based Maps

Consider an environment that contains N_0 landmarks at known positions (x^i_L, y^i_L), $i = 1, \ldots, N_0$. For simplicity, the uncertainties associated with landmark locations are assumed to be zero, although it is relatively straightforward to extend the analysis if this is not the case. At each time step while it is in motion, the robot observes the range (distance) and/or the bearing (relative angle) to one or more landmarks. Observation model provides a mechanism for computing the expected values of observations from sensors, given the knowledge of the map and an estimate

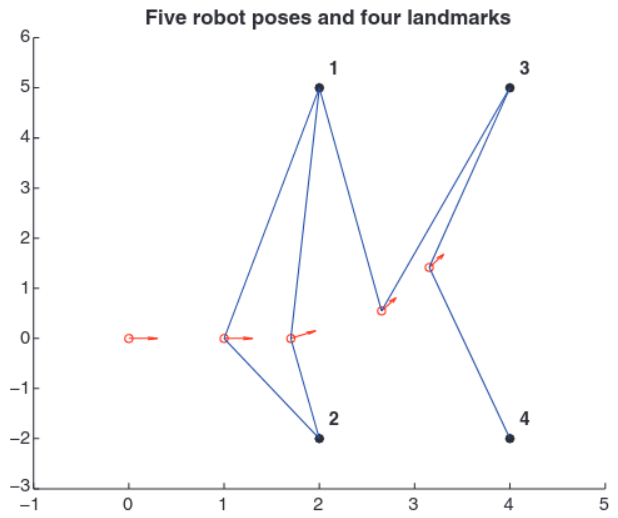

Figure 2. Localization problem with a landmark-based map. The map is defined by four landmarks (black dots) in the environment located at $(2, 5), (2, -2), (4, 5), (4, -2)$, respectively. The robot (the red circle showing the position of robot center, the red arrow showing the orientation) starts from $(0, 0, 0)^T$ at time 0. At time steps 1 and 2, it observes landmarks 1 and 2; at time step 3, it observes landmarks 1 and 3, at time step 4, it observes landmarks 3 and 4. Robot to landmark observations are indicated by blue lines.

of the robot location. If the sensor mounted on the robot observes both the range and the bearing to landmark i at time step $k + 1$, then the observation model is given by

$$\begin{aligned} r^i_{k+1} &= \sqrt{(x^i_L - x^r_{k+1})^2 + (y^i_L - y^r_{k+1})^2} + w_r \\ \theta^i_{k+1} &= a\tan\left(\frac{y^i_L - y^r_{k+1}}{x^i_L - x^r_{k+1}}\right) - \phi^r_{k+1} + w_\theta \end{aligned} \tag{3}$$

where w_r and w_θ are zero-mean Gaussian observation noises.

Laser range finders and ultrasonic sensors are most common sensors used for obtaining range and bearing measurements to landmarks. In case of a sensor that is only able to observe the bearing, for example, a camera, the equation for θ^i_{k+1} becomes the sensor model. A simple robot localization problem with a landmark-based map is illustrated in Figure 2.

2.3. Sensor Model for Occupancy Grid Maps

Occupancy grid maps provide a discretized representation of an environment where each of the grid cells is classified into two categories: occupied or free. Consider the scenario where a sensor on the robot can determine the distance to the nearest occupied grid cell along a given direction. A laser range finder is such a sensor. These sensors consist of a laser beam that rotates at a relatively high speed on the order of tens of revolutions per second and measures the distance to the obstacle that reflects it. If there is no obstacle within the sensor range, a reflection is not received and the sensor typically reports a nominal maximum distance $d_{\max}$. Although the range measurements obtained depend on the environment and the robot location, it is not feasible to find an analytical observation model of the form

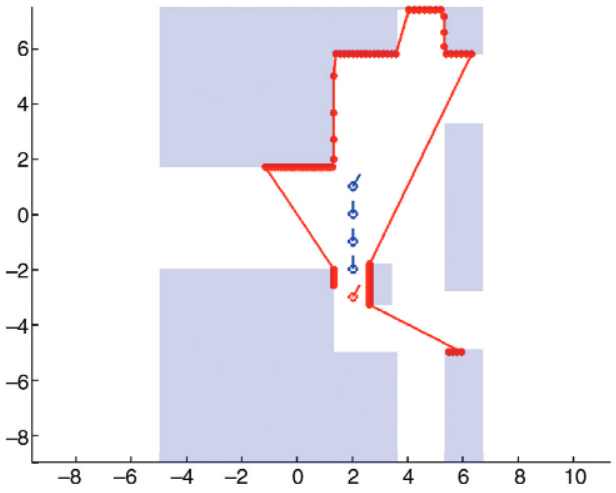

Figure 3. A localization problem with an occupancy grid map: The shaded areas represent occupied cells; the white area represents the free space; the robot moves a few steps (a robot pose is shown as a circle plus an arrow); the readings from the laser range finder at the first pose (red) are depicted as red lines.

equation 3 in this scenario. However, given an estimate of the robot location and the grid map, the expected value of a range measurement can be numerically obtained using ray casting (4). This makes it possible to evaluate the likelihood of a given pose, which is sufficient in some localization approaches such as a particle filter (see Section 4 for details).

Figure 3 shows a simple example of a robot localization problem where a laser range finder observes an environment described using an occupancy grid. The robot moves a few steps in the environment. The scan provided by the sensor at the first pose is shown in red.

3. EXTENDED KALMAN FILTER FOR LOCALIZATION IN LANDMARK-BASED MAPS

The localization problem in a landmark-based map is to find the robot pose at time $k+1$ as

$$\mathbf{x}_{k+1} = (x^r_{k+1}, y^r_{k+1}, \phi^r_{k+1})^T \tag{4}$$

given the map, the sequence of robot actions v_i, ω_i $(i = 0, \ldots, k)$, and sensor observations from time 1 to time $k+1$.

In its most fundamental form, the problem is to estimate the robot poses $\mathbf{x}_i$ $(i = 0, \ldots, k+1)$ that best agree with all robot actions and all sensor observations. This can be formulated as a nonlinear least-squares problem using the motion and observation models derived in Section 2. The solution to the resulting optimization problem can then be calculated using an iterative scheme such as Gauss–Newton to obtain the robot trajectory and as a consequence the current robot pose. Appendix A and Appendix B provide the details on how both linear and nonlinear least-squares problems can be solved, and how the localization problem can be formulated as a nonlinear least-squares problem. The dimensionality of the problem is $3(k+1)$ for two-dimensional motion, and given the sampling rate of modern sensors are on the order of tens of hertz, this strategy quickly becomes computationally intractable.

If the noises associated with the sensor measurements can be approximated using Gaussian distributions, and an initial estimate for the robot location at time 0, described using a Gaussian distribution $\mathbf{x}_0 \sim N(\hat{\mathbf{x}}_0, P_0)$ with known $\hat{\mathbf{x}}_0, P_0$ is available (in this article, $\hat{\mathbf{x}}$ is used to denote the estimated value of $\mathbf{x}$), an approximate solution to this nonlinear least-squares problem can be obtained using an EKF. EKF effectively summarizes all the measurements obtained in the past in the estimate of the current robot location and its covariance matrix. When a new observation from the sensor becomes available, the current robot location estimate and its covariance are updated to reflect the new information gathered. Essential steps of the EKF-based localization algorithm are described in the following:

Let us denote

$$\mathbf{u}_k = (v_k, \omega_k)^T, \ \mathbf{w}_k = (\delta v, \delta\omega)^T. \tag{5}$$

Then the nonlinear process model (from time k to time $k+1$) as stated in equation 2 can be written in a compact form as

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \tag{6}$$

where $\mathbf{f}$ is the system transition function, $\mathbf{u}_k$ is the control, and $\mathbf{w}_k$ is the zero-mean Gaussian process noise $\mathbf{w}_k \sim N(0, Q)$.

Consider the general case where more than one landmark is observed. Representing all the observations $r^i_{k+1}, \theta^i_{k+1}$ together as a single vector $\mathbf{z}_{k+1}$, and all the noises w_r, w_θ together as a single vector $\mathbf{v}_{k+1}$, the observation model at time $k+1$ as stated in equation 3 can also be written in a compact form as

$$\mathbf{z}_{k+1} = \mathbf{h}(\mathbf{x}_{k+1}) + \mathbf{v}_{k+1} \tag{7}$$

where $\mathbf{h}$ is the observation function obtained from equation 3 and $\mathbf{v}_{k+1}$ is the zero-mean Gaussian observation noise $\mathbf{v}_{k+1} \sim N(0, R)$.

Let the best estimate of $\mathbf{x}_k$ at time k be

$$\mathbf{x}_k \sim N(\hat{\mathbf{x}}_k, P_k). \tag{8}$$

Then the localization problem becomes one of estimating $\mathbf{x}_{k+1}$ at time $k+1$:

$$\mathbf{x}_{k+1} \sim N(\hat{\mathbf{x}}_{k+1}, P_{k+1}) \tag{9}$$

where $\hat{\mathbf{x}}_{k+1}, P_{k+1}$ are updated using the information gathered using the sensors. EKF framework achieves this as follows. To maintain clarity, only the basic equations are presented in the following, while Appendix C provides a more detailed explanation.

Predict using process model:

$$\bar{\mathbf{x}}_{k+1} = \mathbf{f}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) \tag{10}$$

$$\begin{aligned}\bar{P}_{k+1} = {} & J_{\mathbf{f}_x}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) P_k J^T_{\mathbf{f}_x}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) \\ & + J_{\mathbf{f}_w}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) Q J^T_{\mathbf{f}_w}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0)\end{aligned} \tag{11}$$

where $J_{\mathbf{f}_x}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0)$ is the Jacobian of function $\mathbf{f}$ with respect to $\mathbf{x}$, $J_{\mathbf{f}_w}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0)$ is the Jacobian of function $\mathbf{f}$ with respect to $\mathbf{w}$, both evaluated at $(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0)$.

Update using observation:

$$\hat{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_{k+1} + K(\mathbf{z}_{k+1} - \mathbf{h}(\bar{\mathbf{x}}_{k+1})) \quad (12)$$

$$P_{k+1} = \bar{P}_{k+1} - KSK^T \quad (13)$$

where the innovation covariance S (here $\mathbf{z}_{k+1} - \mathbf{h}(\bar{\mathbf{x}}_{k+1})$ is called innovation) and the Kalman gain K are given by

$$S = J_{\mathbf{h}}(\bar{\mathbf{x}}_{k+1})\bar{P}_{k+1}J_{\mathbf{h}}^T(\bar{\mathbf{x}}_{k+1}) + R \quad (14)$$

$$K = \bar{P}_{k+1}J_{\mathbf{h}}^T(\bar{\mathbf{x}}_{k+1})S^{-1} \quad (15)$$

where $J_{\mathbf{h}}(\bar{\mathbf{x}}_{k+1})$ is the Jacobian of function $\mathbf{h}$ with respect to $\mathbf{x}$ evaluated at $\bar{\mathbf{x}}_{k+1}$.

Recursive application of the above equations every instant a new observation is gathered yields an updated estimate for the current robot location and its uncertainty. This recursive nature makes EKF the most computationally efficient algorithm available for robot localization.

An important prerequisite for EKF-based localization is the ability to associate measurements obtained with specific landmarks present in the environment. Landmarks may be artificial, for example, laser reflectors, or natural geometric features present in the environment such as line segments, corners, or planes (5,6). In many cases, the observation itself does not contain any information as to which particular landmark is being observed. Data association is the process in which a decision is made as to the correspondence between an observation from the sensor and a particular landmark. Data association is critical to the operation of an EKF-based localizer, as catastrophic failure may result if data association decisions are incorrect.

EKF relies on approximating the nonlinear motion and observation models using linear equations and that the sensor noises can be approximated using Gaussian distributions. These are reasonable assumptions under many practical conditions and therefore EKF is the obvious choice for solving the robot localization problem when the map of the environment consists of clearly identifiable landmarks.

Figure 4 shows the result of EKF localization for the simple problem given in Figure 2. The ground truth of the robot poses and the estimated robot poses are shown in red and blue, respectively. The 95% confidence ellipses obtained from the covariance matrices in the EKF estimation process are also shown in the figure.

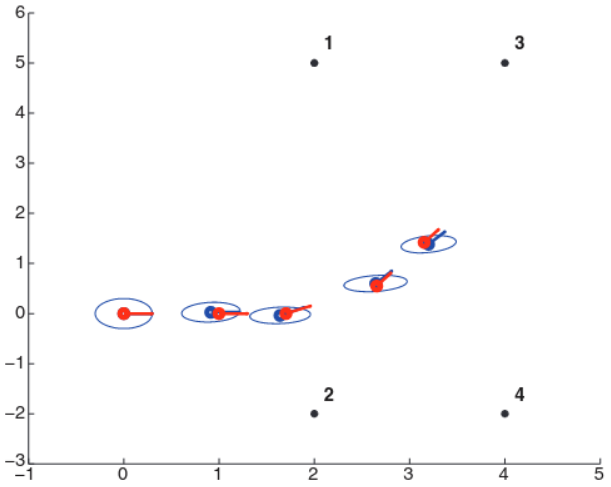

Figure 4. Result of EKF localization for the simple problem given in Figure 2. The ground truth of the robot poses and the estimated robot poses by EKF are shown in red and blue, respectively. The ellipses are the 95% confidence ellipses obtained from the covariance matrices in the EKF estimation result.

4. PARTICLE FILTER FOR LOCALIZATION IN GRID MAPS

There are two important situations where EKF is not the method of choice for robot localization. The first is when the environment is represented by an occupancy grid. Sensor model for occupancy grid maps described in Section 2.3 is not an analytic model but based on the numerical process of ray casting and as such is unsuitable for use with an EKF. The other situation is when initial robot location is completely unknown, usually known as the global localization problem. In this case, the location of the robot needs to be described using an arbitrary probability distribution; thus, the Gaussian assumption that is the basis of the EKF formulation is violated. In general, manipulating arbitrary probability distributions is computationally intractable. One possible strategy is to discretize the space of possible robot locations and thus deal with manipulating discrete probability distribution. This method is known as Markov localization. The computation burden associated with Markov localization is proportional to the size of the environment and the resolution of the discretization, making this strategy unsuitable in many situations. Markov localization is described in Appendix D.

In the particle filter localization (also known as Monte Carlo localization) (7), rather than discretizing the space of robot locations, a weighted set of robot location estimates, termed as particles, is used to describe the probability distribution of the robot location. As the computations are focused on particles and more particles are placed at more probable robot locations, the particle filters provide a more efficient alternative to Markov localization. Number of particles used determines the accuracy of the representation. However, increasing the number of particles to obtain a higher accuracy leads to a more costly estimation process.

In the particle filter, each particle in effect provides a guess as to the location of the robot. Thus, each particle is represented by three variables (x, y (position), ϕ (orientation)) for a robot operating in a two-dimensional plane. Each particle i has a weight w_i that indicates the contribution of that particular particle to the probability distribution. The sum of the weights of all particles is set to 1, that is, $\sum_{i=1}^{n} w_i = 1$, where n is the total number of particles used. A collection of such guesses describes the best knowledge available, usually termed the belief, as to the true location of the robot. In the case of global localization, the initial robot location is completely unknown; therefore, all locations of the environment are equally likely to contain the robot. Thus, a set of equally weighted particles uniformly distributed in the environment is used to represent the belief of the robot location. During the localization

process, this belief is updated as more and more information is acquired from the sensors.

In the particle filter, every time information from the sensors is gathered, the current belief is updated. The process is as follows (8,9):

(i) *Prediction:* When the robot is commanded to move, the new belief is obtained by moving each particle according to the motion model equation 2 with randomly generated $\delta v_k, \delta\omega_k$.

(ii) *Update:* When a new sensor observation is received, the belief is updated using an observation model. In this step, the weights of the particles are changed to reflect the likelihood that the true robot location coincides with the corresponding particle. In the case of j^{th} observation from a laser range finder, ray casting from each particle is used to obtain an expected measurement $\hat{d}_j$. If the actual measurement is given by d_j and if the sensor noise is assumed to be zero mean with a variance σ_d^2, the likelihood can be computed using a Gaussian distribution based on

$$\frac{1}{\sigma_d\sqrt{2\pi}}\ \exp\left\{-\frac{(\hat{d}_j - d_j)^2}{2\sigma_d^2}\right\}. \tag{16}$$

As at a given instance multiple independent range observations are acquired from the sensor, likelihood of obtaining a sequence of observations is computed by multiplying together all the likelihood. Once likelihoods of all the particles are computed, these are normalized to obtain the weight of each of the particles.

(iii) *Resampling*: This is performed to avoid the situation where a small number of particles with large weights dominate the representation of the belief. One common strategy used for resampling (10) is as follows:

(a) Compute an estimate of the effective number of particles as

$$n_{\text{eff}} = \frac{1}{\sum_{i=1}^{n} w_i^2}. \tag{17}$$

(b) If n_{eff} is less than a threshold, then draw n particles from the current particle set with probabilities proportional to their weights. Replace the current particle set with this new one. Set the weights of each particle to be $1/n$.

(iv) Resulting set of particles represents the updated belief of the robot location.

This process is repeated as new control actions are taken and new observations become available. The mean or the mode of the corresponding probability distribution can be used if a numerical value for the best estimate of the robot location is desired.

Figure 5 shows the result of particle filter localization for the simple problem given in Figure 3. The particles, the best estimate of the robot location, and the ground truth robot location at the last step are shown in green, blue, and red, respectively.

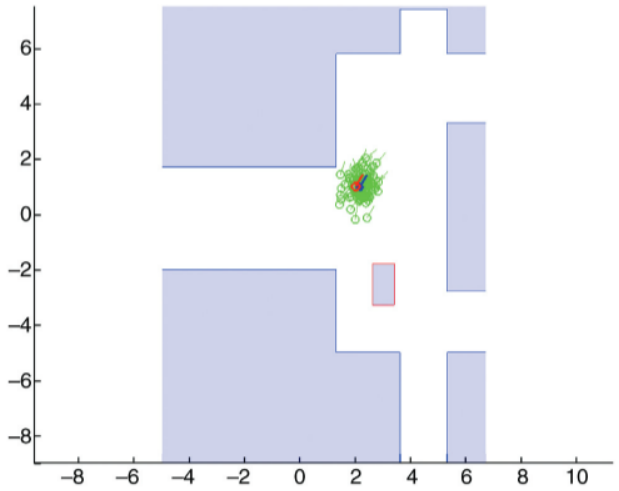

Figure 5. The result of particle filter localization: the particles in the last step are shown in green, the ground truth of the last robot pose is in red, the best estimate of the last robot pose is in blue.

5. ALTERNATIVE LOCALIZATION TECHNIQUES

When a map of the environment is not available, the robot localization problem becomes significantly more challenging. In this case, the problem becomes one of estimating both the robot location and landmark locations simultaneously. In the past 15 years, robust and efficient methods to dealing with robot localization in unknown environments, also known as the simultaneous localization and mapping (SLAM) problem (3), have emerged. SLAM in a feature-based environment has been well studied and it has been shown that both EKF and least-squares optimization could be used to reliably solve SLAM in most cases (11,12). Presence of efficient solvers for very large-scale nonlinear least-squares optimization problems (13) has resulted in real-time solutions to the robot location estimation in unknown environments. Due to their low cost and presence of algorithms that extract rich information, cameras have become an important sensor in many robot navigation applications. A comprehensive survey on vision-based navigation is reported in Reference 14, while a recent strategy for SLAM using information from a camera is reported in Reference 15. Combining information from monocular cameras with inertial sensors makes it possible to obtain reliable localization in real time with a high level of accuracy (16).

An alternative method for localization in unknown environments is to simply use the information from a sensor such as a laser range finder to obtain the translation and rotation relating two robot poses from which two laser scans are taken. This is done through scan matching that aligns the two scans. Collection of such relative pose estimates can be used to formulate a nonlinear least-squares problem that can be solved to estimate all the robot poses (17). A modern solver such as g2o (13) could be used to efficiently obtain the location estimates. A representation of the environment (map) is not estimated. This strategy is called pose-graph SLAM (18). For a feature-based or

pose-graph SLAM technique to be effective, a strategy to recognize the fact that the robot has returned to areas it has visited before, known as loop closure, is required. The reliable detection of loop closure is one of the remaining challenges for both feature-based SLAM and pose-graph SLAM (19).

Many other interesting robot localization problems and alternative solutions have been reported in the literature. Typical examples are localization using signal strengths from wireless infrastructure (20) and RFIDs (21), fusion of information from GPS and inertial navigation units (22), and an optimization technique for localization in grid maps (23) based on chamfer distance (24) that does not rely on many tuning parameters as is usually the case with particle filters.

The problem of estimating robot location given a map is now considered a solved problem, although highly dynamic environments populated with people pose significant challenges in practical deployments. The more complex problem of continuously estimating robot location within an unknown environment over a long period is still the subject of much research.

BIBLIOGRAPHY

1. G. Georges, R. Sobek, and R. Chatila. A Multi-Level Planning and Navigation System for a Mobile Robot: A First Approach to Hilare. In *Proc. 6th International Joint Conference on Artificial Intelligence*; Vol. 1, Morgan Kaufmann Publishers Inc., 1979.
2. H. F. Durrant-Whyte. *Ind. Rob.* **1994**, *21*, pp 11–16.
3. S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.
4. H. Blume and B. Heimann. A Laser Range Scanner Simulation for Probabilistic Object Tracking, in *ICRA 2007 Workshop: Planning, Perception and Navigation for Intelligent Vehicles*; 2007.
5. M. Drumheller. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *2*, pp 325–332.
6. J. J. Leonard and H. F. Durrant-Whyte. *IEEE Trans. Rob. Autom.* **1991**, *7*, pp 376–382.
7. I. M. Rekleitis. A Particle Filter Tutorial for Mobile Robot Localization. Centre for Intelligent Machines, Technical Report TR-CIM-04-02, 2004), McGill University.
8. F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo Localization for Mobile Robots. *Proc. IEEE International Conference on Robotics and Automation*; Vol. 2. IEEE, 1999.
9. D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *Proc. Sixteenth National Conference on Artificial Intelligence*; John Wiley & Sons Ltd, 1999.
10. G. Kitagawa. *J. Comput. Graph Stat.* **1996**, *5*, pp 1–25.
11. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. *IEEE Trans. Rob. Autom.* **2001**, *17*, pp 229–241.
12. F. Dellaert and M. Kaess. *Int. J. Rob. Res.* **2006**, *25*(12), pp 1181–1203.
13. R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A General Framework for Graph Optimization. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*; 2011, pp 3607–3613.
14. G. N. DeSouza, and C. K. Avinash. *Pattern Anal. Mach. Intell.* **2002**, *24*, pp 237–267.
15. L. Zhao, S. Huang, Y. Sun, L. Yan, and G. Dissanayake. *Int. J. Rob. Res.* **2015**, 34(4-5), pp 493–516.
16. J. Hesch, D. G. Kottas, S. L. Bowman, and S. Roumeliotis. *IEEE Trans. Rob.* **2014**, *30*, pp 158–176.
17. F. Lu and E. Milios. *Auton. Robots* **1997**, 4, pp 333–349.
18. G. Grisetti, R. Kmmerle, C. Stachniss, and W. Burgard. *IEEE Intell. Trans. Syst. Mag.* **2010**, *2*, pp 31–43.
19. Y. Latif, C. Cadena, and J. Neira. *Int. J. Rob. Res.* **2013**, *32*(14), pp 1611–1626.
20. B. Ferris, D. Haehnel, and D. Fox. Gaussian Processes for Signal Strength-Based Location Estimation. In *Proc. Robotics: Science and Systems*; 2006.
21. J. Zhou and J. Shi. *J. Intell. Manuf.* **2009**, *20*, pp 695–707.
22. S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte. *IEEE Trans. Rob. Autom.* **1999**, *15*, pp 572–578.
23. L. Dantanarayana, R. Ranasinghe, and G. Dissanayake. C-LOG: A Chamfer Distance Based Method for Localisation in Occupancy Grid-Maps. In *2013 IEEE / RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 376–381.
24. G. Borgefors. *Comput. Vis. Graph. Image Process.* **1986**, *34*, pp 344–371.

APPENDIX A: LEAST SQUARES PROBLEM AND GAUSS–NEWTON ITERATION

A.1 Linear Least Squares Problem and Solution

A linear least squares problem is to find $\mathbf{X}$ that minimizes

$$\|\mathbf{b} - A\mathbf{X}\|^2 \stackrel{\Delta}{=} (\mathbf{b} - A\mathbf{X})^T(\mathbf{b} - A\mathbf{X})$$

for given $\mathbf{b}$, A, where

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad m \geq n$$

and

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}.$$

Assuming that the matrix A is full column rank, the above problem has a closed-form solution:

$$\mathbf{X}^* = (A^T A)^{-1} A^T \mathbf{b}. \tag{18}$$

This can be obtained by expanding the objective function

$$\begin{aligned} E(\mathbf{X}) &= \|\mathbf{b} - A\mathbf{X}\|^2 = (\mathbf{b} - A\mathbf{X})^T(\mathbf{b} - A\mathbf{X}) \\ &= \mathbf{b}^T\mathbf{b} - 2\mathbf{b}^T A\mathbf{X} + \mathbf{X}^T A^T A\mathbf{X} \end{aligned}$$

and finding stationary points by letting

$$\frac{\mathrm{d}E}{\mathrm{d}\mathbf{X}} = 2A^T A\mathbf{X} - 2A^T\mathbf{b} = 0$$

which leads to the solution equation 18.

A.2 Weighted Linear Least Squares Problem and Solution

A weighted linear least squares problem is to find $\mathbf{X}$ that minimizes

$$(\mathbf{b} - A\mathbf{X})^T P^{-1}(\mathbf{b} - A\mathbf{X})$$

where P is a positive definite matrix.

The closed-form solution to the weighted linear least-squares problem is

$$\mathbf{X}^* = (A^T P^{-1} A)^{-1} A^T P^{-1}\mathbf{b}. \tag{19}$$

A.3 Nonlinear Least Squares and Gauss–Newton Iteration

A nonlinear least squares problem is to find $\mathbf{X}$ that minimizes

$$\|\mathbf{Z} - \mathbf{F}(\mathbf{X})\|^2 = [\mathbf{Z} - \mathbf{F}(\mathbf{X})]^T[\mathbf{Z} - \mathbf{F}(\mathbf{X})]$$

where $\mathbf{Z} = [z_1, z_2, \dots, z_m]^T$, $\mathbf{X} = [x_1, x_2, \dots, x_n]^T$,

$$\mathbf{F}(\mathbf{X}) = \begin{bmatrix} f_1(\mathbf{X}) \\ f_2(\mathbf{X}) \\ \vdots \\ f_m(\mathbf{X}) \end{bmatrix} = \begin{bmatrix} f_1(x_1, x_2, \dots, x_n) \\ f_2(x_1, x_2, \dots, x_n) \\ \vdots \\ f_m(x_1, x_2, \dots, x_n) \end{bmatrix}$$

with $m \geq n$.

In general, closed-form solution for a nonlinear least squares problem cannot be obtained. Many techniques for solving nonlinear least squares problems are based on iteration.

Suppose $\mathbf{X}$ is close to $\mathbf{X}_0$, by linearization,

$$\mathbf{F}(\mathbf{X}) \approx \mathbf{F}(\mathbf{X}_0) + J_{\mathbf{F}}(\mathbf{X}_0)(\mathbf{X} - \mathbf{X}_0)$$

where $J_{\mathbf{F}}(\mathbf{X}_0)$ is the Jacobian matrix given by

$$J_{\mathbf{F}}(\mathbf{X}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

evaluated at $\mathbf{X}_0$.

Thus,

$$\mathbf{Z} - \mathbf{F}(\mathbf{X}) \approx \mathbf{Z} - \mathbf{F}(\mathbf{X}_0) + J_{\mathbf{F}}(\mathbf{X}_0)\mathbf{X}_0 - J_{\mathbf{F}}(\mathbf{X}_0)\mathbf{X}.$$

Let

$$A = J_{\mathbf{F}}(\mathbf{X}_0), \quad \mathbf{b} = \mathbf{Z} - \mathbf{F}(\mathbf{X}_0) + J_{\mathbf{F}}(\mathbf{X}_0)\mathbf{X}_0.$$

Assuming that the matrix $J_{\mathbf{F}}(\mathbf{X}_0)$ is full column rank, using the linear least square solution equation 18, we get

$$\begin{aligned} \mathbf{X}_1 = {} & [J_{\mathbf{F}}^T(\mathbf{X}_0) J_{\mathbf{F}}(\mathbf{X}_0)]^{-1} J_{\mathbf{F}}^T(\mathbf{X}_0) \\ & [\mathbf{Z} - \mathbf{F}(\mathbf{X}_0) + J_{\mathbf{F}}(\mathbf{X}_0)\mathbf{X}_0]. \end{aligned} \tag{20}$$

In general, the iteration step is

$$\begin{aligned} \mathbf{X}_{k+1} = {} & [J_{\mathbf{F}}^T(\mathbf{X}_k) J_{\mathbf{F}}(\mathbf{X}_k)]^{-1} J_{\mathbf{F}}^T(\mathbf{X}_k) \\ & [\mathbf{Z} - \mathbf{F}(\mathbf{X}_k) + J_{\mathbf{F}}(\mathbf{X}_k)\mathbf{X}_k]. \end{aligned} \tag{21}$$

Iterating until convergence leads to the optimum solution, provided that the initial guess $\mathbf{X}_0$ is sufficiently close to the solution. This is known as the Gauss–Newton iteration.

A.4 Weighted Nonlinear Least Squares Problem

The weighted nonlinear least squares problem is to find $\mathbf{X}$ that minimize

$$[\mathbf{Z} - \mathbf{F}(\mathbf{X})]^T P^{-1}[\mathbf{Z} - \mathbf{F}(\mathbf{X})] \tag{22}$$

where P is the covariance matrix of the noises contained in the measurement (data) $\mathbf{Z}$.

Given the initial value $\mathbf{X}_0$, Gauss–Newton iteration step is

$$\begin{aligned} \mathbf{X}_{k+1} = {} & [J_{\mathbf{F}}^T(\mathbf{X}_k) P^{-1} J_{\mathbf{F}}(\mathbf{X}_k)]^{-1} \\ & J_{\mathbf{F}}^T(\mathbf{X}_k) P^{-1}[\mathbf{Z} - \mathbf{F}(\mathbf{X}_k) + J_{\mathbf{F}}(\mathbf{X}_k)\mathbf{X}_k]. \end{aligned} \tag{23}$$

APPENDIX B: LEAST SQUARES METHOD FOR LANDMARK BASED LOCALIZATION

Least squares method provides a way to optimally estimate the robot trajectory using all the available robot motion and observation information up to time $k+1$. In least squares method, we estimate all the robot poses together instead of only the last pose as in the case of both the EKF and the particle filter based robot localization.

The state vector in least squares method is

$$\mathbf{X} = (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{x}_{k+1})^T. \tag{24}$$

Note that the initial state mean value can be expressed as an observation of $\mathbf{x}_0$:

$$\hat{\mathbf{x}}_0 = \mathbf{x}_0 + \mathbf{v}_{\mathbf{x}_0} \tag{25}$$

where $\mathbf{v}_{\mathbf{x}_0} \sim N(0, P_0)$ is the "observation noise".

The motion model equation 2 can be rewritten as

$$\begin{aligned} v_k &= \frac{\sqrt{(x^r_{k+1} - x^r_k)^2 + (y^r_{k+1} - y^r_k)^2}}{\Delta T} - \delta v_k \\ \omega_k &= \frac{\phi^r_{k+1} - \phi^r_k}{\Delta T} - \delta\omega_k. \end{aligned} \tag{26}$$

Now all the information available from time 0 to time $k+1$ is summarized in

$$\begin{aligned} \mathbf{Z} = {} & (\hat{\mathbf{x}}_0^T, v_0, w_0, \dots, r^i_1, \theta^i_1, \dots, \\ & v_k, w_k, \dots, r^i_{k+1}, \theta^i_{k+1}, \dots)^T. \end{aligned} \tag{27}$$

The relation between $\mathbf{Z}$ and $\mathbf{X}$ is given by

$$\mathbf{Z} = \mathbf{F}(\mathbf{X}) + \mathbf{V} \tag{28}$$

where $\mathbf{F}(\mathbf{X})$ is the nonlinear function combining equations 3, 25, and 26. $\mathbf{V} \sim N(0, P)$ is the vector of all the noises where P is constructed by P_0, Q, R.

Now the localization problem can be formulated as a nonlinear least squares problem as in equation 22 and could be solved using Gauss–Newton iteration by equation 23.

APPENDIX C: INTUITIVE EXPLANATIONS OF THE KALMAN FILTER

C.1 Kalman Filter Equations

When both the process model and the observation model are linear, they can be expressed by

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + G\mathbf{u}_k + \mathbf{w}_k \tag{29}$$

and

$$\mathbf{z}_{k+1} = H\mathbf{x}_{k+1} + \mathbf{v}_{k+1} \tag{30}$$

where $\mathbf{x}_k, \mathbf{x}_{k+1}$ are the system state at time $k, k+1$, F is the system transition matrix, G is the gain of control $\mathbf{u}_k$, and $\mathbf{w}_k$ is the zero-mean Gaussian process noise $\mathbf{w}_k \sim N(0, Q)$, H is the observation matrix, and $\mathbf{v}_{k+1}$ is the zero-mean Gaussian observation noise $\mathbf{v}_{k+1} \sim N(0, R)$.

In this case, the state estimation can be obtained using the following Kalman filter equations assuming the initial state $\mathbf{x}_0$ follows a known Gaussian distribution $\mathbf{x}_0 \sim N(\hat{\mathbf{x}}_0, P_0)$.

Predict using process model:

$$\bar{\mathbf{x}}_{k+1} = F\hat{\mathbf{x}}_k + G\mathbf{u}_k \tag{31}$$

$$\bar{P}_{k+1} = FP_kF^T + Q \tag{32}$$

where $\bar{\mathbf{x}}_{k+1}$ is the state estimate at time $k+1$ before using the observation information at time $k+1$, and $\bar{P}_{k+1}$ is its corresponding covariance matrix.

Update using observation:

$$\hat{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_{k+1} + K(\mathbf{z}_{k+1} - H\bar{\mathbf{x}}_{k+1}) \tag{33}$$

$$P_{k+1} = \bar{P}_{k+1} - KSK^T \tag{34}$$

where the innovation covariance S (here $\mathbf{z}_{k+1} - H\bar{\mathbf{x}}_{k+1}$ is called innovation) and the Kalman gain K are given by

$$S = H\bar{P}_{k+1}H^T + R \tag{35}$$

$$K = \bar{P}_{k+1}H^TS^{-1}. \tag{36}$$

In the following, we will provide some intuitive explanations of the Kalman filter equations for the one-dimensional case.

C.2 One-Dimensional Gaussian Distribution and Its Information

If a random variable x follows a Gaussian distribution, it is denoted as

$$x \sim N(m, \sigma^2) \tag{37}$$

where m is the mean and σ^2 is the variance.

(Fisher) information of a Gaussian distribution $N(m, \sigma^2)$ is the inverse of the variance,

$$I = \frac{1}{\sigma^2}. \tag{38}$$

Intuitively speaking, the larger the uncertainty, the smaller the information.

C.3 Important Properties of 1D Gaussian Distributions

The following are a few properties of 1D Gaussian distributions that are useful in deriving the Kalman filter equations:

- For any constant a,

$$x \sim N(m, \sigma^2) \Rightarrow ax \sim N(am, a^2\sigma^2). \tag{39}$$

- For any constant u,

$$x \sim N(m, \sigma^2) \Rightarrow x + u \sim N(m + u, \sigma^2). \tag{40}$$

- For two independent random variables x and y,

$$\begin{aligned} &x \sim N(m_x, \sigma_x^2), y \sim N(m_y, \sigma_y^2) \\ \Rightarrow\ &x + y \sim N(m_x + m_y, \sigma_x^2 + \sigma_y^2). \end{aligned} \tag{41}$$

C.4 One-Dimensional Kalman Filter Prediction

This section shows that the Kalman filter prediction equation can be obtained easily from the properties of 1D Gaussian distributions listed in Section C.3.

Suppose the process model is

$$x_{k+1} = x_k + u_k + w_k \tag{42}$$

where u_k is the control and w_k is the zero-mean Gaussian process noise with variance σ_u^2. That is, $w_k \sim N(0, \sigma_u^2)$. It is also assumed that w_k is independent of x_k.

At time k, the estimate of x_k follows a Gaussian distribution $x_k \sim N(\hat{x}_k, \sigma_k^2)$ (see equation 8), thus, by equation 40,

$$x_k + u_k \sim N(\hat{x}_k + u_k, \sigma_k^2). \tag{43}$$

Further by equation 41,

$$x_{k+1} = (x_k + u_k) + w_k \sim N(\hat{x}_k + u_k, \sigma_k^2 + \sigma_u^2). \tag{44}$$

Thus, if we denote the estimate of x_{k+1} (after the process but before the observation) as

$$x_{k+1} \sim N(\bar{x}_{k+1}, \bar{\sigma}_{k+1}^2) \tag{45}$$

then the prediction equations are

$$\begin{aligned} \bar{x}_{k+1} &= \hat{x}_k + u_k \\ \bar{\sigma}_{k+1}^2 &= \sigma_k^2 + \sigma_u^2. \end{aligned} \tag{46}$$

Similarly, if the process model is

$$x_{k+1} = ax_k + bu_k + w_k \tag{47}$$

where a, b are constants and $w_k \sim N(0, \sigma_u^2)$, then the prediction equations become

$$\begin{aligned} \bar{x}_{k+1} &= a\hat{x}_k + bu_k \\ \bar{\sigma}_{k+1}^2 &= a^2\sigma_k^2 + \sigma_u^2 \end{aligned} \tag{48}$$

which are equations 31 and 32 when all variables are scalars.

C.5 One-Dimensional Kalman Filter Update

Suppose the observation model is

$$z_{k+1} = x_{k+1} + v_{k+1} \tag{49}$$

where z_{k+1} is the observation value at time $k+1$ and v_{k+1} is the zero-mean Gaussian observation noise with variance σ_z^2. That is, $v_{k+1} \sim N(0, \sigma_z^2)$. It is also assumed that v_{k+1} is independent of x_{k+1}. By equation 39 (choosing $a = -1$),

$$-v_{k+1} \sim N(0, \sigma_z^2). \tag{50}$$

By the observation model equation 49,

$$x_{k+1} = -v_{k+1} + z_{k+1}. \tag{51}$$

Thus, by equation 40,

$$x_{k+1} \sim N(z_{k+1}, \sigma_z^2). \tag{52}$$

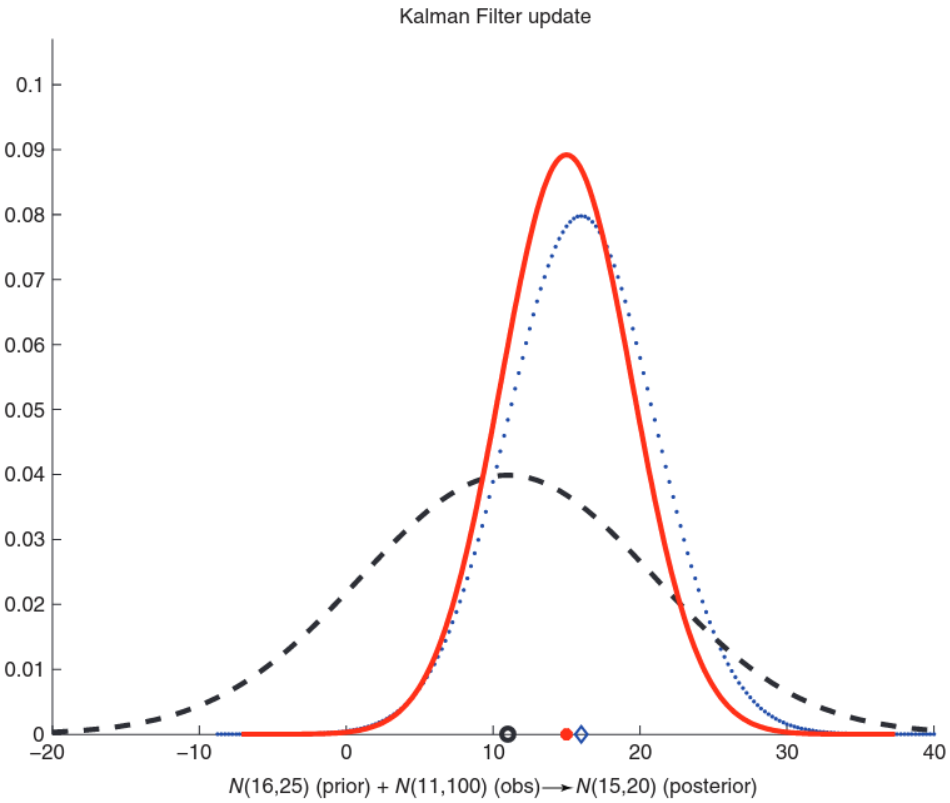

Figure 6. One-dimensional Kalman filter update (dot line is the prior, dashed line is the observation, solid line is the posterior). $I_{\text{prior}} = 1/25$, $I_{\text{obs}} = 1/100$, $I_{\text{total}} = I_{\text{prior}} + I_{\text{obs}} = 1/20$; posterior variance $= \frac{1}{I_{\text{total}}} = 20$; posterior mean $= \frac{I_{\text{prior}}}{I_{\text{total}}} \times 16 + \frac{I_{\text{obs}}}{I_{\text{total}}} \times 11 = 15$.

The prior information about x_{k+1} is given by equation 45 (after the prediction but before the update). So we have two pieces of information about x_{k+1}: one from observation equation 52 and one from prior equation 45.

According to the definition of information contained in a Gaussian distribution (see equation 38), the information (about x_{k+1}) contained in equation 45 is

$$I_{\text{prior}} = \frac{1}{\bar{\sigma}^2_{k+1}} \tag{53}$$

while the information (about x_{k+1}) contained in equation 52 is

$$I_{\text{obs}} = \frac{1}{\sigma_z^2}. \tag{54}$$

The total information (about x_{k+1}) after the observation should be the sum of the two, namely,

$$I_{\text{total}} = I_{\text{prior}} + I_{\text{obs}} = \frac{1}{\bar{\sigma}^2_{k+1}} + \frac{1}{\sigma_z^2}. \tag{55}$$

The new mean value is the weighted sum of the mean values of the two Gaussian distributions equations 45 and 52. The weights are decided by the proportion of information contained in each of the Gaussian distributions (as compared with the total information). That is,

$$\begin{aligned} \hat{x}_{k+1} &= \frac{I_{\text{prior}}}{I_{\text{total}}} \bar{x}_{k+1} + \frac{I_{\text{obs}}}{I_{\text{total}}} z_{k+1} \\ &= \frac{\sigma_z^2}{\bar{\sigma}^2_{k+1}+\sigma_z^2} \bar{x}_{k+1} + \frac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2} z_{k+1}. \end{aligned} \tag{56}$$

Note that the sum of the two weights is 1, that is, $\frac{I_{\text{prior}}}{I_{\text{total}}} + \frac{I_{\text{obs}}}{I_{\text{total}}} = 1$.

The variance can be obtained by (see equation 38)

$$\sigma^2_{k+1} = \frac{1}{I_{\text{total}}} = \frac{1}{1/\bar{\sigma}^2_{k+1} + 1/\sigma_z^2} = \frac{\bar{\sigma}^2_{k+1}\sigma_z^2}{\bar{\sigma}^2_{k+1} + \sigma_z^2}. \tag{57}$$

So, the final estimate on x_{k+1} (after the prediction and update) is

$$x_{k+1} \sim N(\hat{x}_{k+1}, \sigma^2_{k+1}) \tag{58}$$

where $\hat{x}_{k+1}$ and σ^2_{k+1} are given in equations 56 and 57, respectively.

Figure 6 illustrates the update step of Kalman filter.

The update formulas, equations 56 and 57, can also be expressed as

$$\begin{aligned} \hat{x}_{k+1} &= \bar{x}_{k+1} + \tfrac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2}(z_{k+1} - \bar{x}_{k+1}) \\ \sigma^2_{k+1} &= \bar{\sigma}^2_{k+1}(1 - \tfrac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2}) \\ &= \bar{\sigma}^2_{k+1} - \bar{\sigma}^2_{k+1}\tfrac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2} \\ &= \bar{\sigma}^2_{k+1} - \tfrac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2}(\bar{\sigma}^2_{k+1} + \sigma_z^2)\tfrac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2}. \end{aligned} \tag{59}$$

These are equations 33 and 34 (when all variables are scalars), where $z_{k+1} - \bar{x}_{k+1}$ is the innovation, $\bar{\sigma}^2_{k+1} + \sigma_z^2$ is the innovation variance S, and $\frac{\bar{\sigma}^2_{k+1}}{\bar{\sigma}^2_{k+1}+\sigma_z^2}$ is the Kalman gain K.

APPENDIX D: MARKOV LOCALIZATION

In Makov localization, rather than relying on a set of particles to represent a probability distribution, the state space is discretized into a grid and the probability that the robot is present in a particular grid cell is used to describe the estimate of the robot location. In a two-dimensional scenario, the discretization is over three–dimensional space incorporating the robot position and orientation.

At time k, the probability that the robot is present in each of the grid cells is represented by

$$p_i(k) \triangleq P(\mathbf{x}_k = i), \quad i = 1, \dots, M \tag{60}$$

where $P(\mathbf{x}_k = i)$ means the probability of robot pose $\mathbf{x}_k$ is in grid cell i, M is the total number of grid cells, and

$$0 \le p_i(k) \le 1, \quad \sum_{i=1}^{M} p_i(k) = 1.$$

This probability distribution is called belief $\mathrm{bel}(k)$.

Initial belief $\mathrm{bel}(0)$ is the prior distribution. When there is no prior knowledge about the robot location, the probability distribution is a uniform one. That is,

$$p_i(0) = \frac{1}{M}, \quad i = 1, \dots, M.$$

Given a belief $\mathrm{bel}(k)$ at time k and a new control input $\mathbf{u}_k$ and a new observation $\mathbf{z}_{k+1}$, the belief needs to be updated to find $\mathrm{bel}(k+1)$ using Bayes filter. There are two essential steps used to update the belief:

i. *Prediction*: In this step the new control input $\mathbf{u}_k$ and the previous belief $\mathrm{bel}(k)$ are used to compute the predicted belief $\overline{\mathrm{bel}(k+1)}$. Here $\overline{\mathrm{bel}(k+1)}$ is the *prediction*, which for any possible location j can be computed using

$$\overline{p_j(k+1)} = \sum_{i=1}^{M} p_i(k) P(\mathbf{x}_{k+1} = j | \mathbf{x}_k = i, \mathbf{u}_k). \tag{61}$$

This equation is obtained using the law of total probabiliy. Here $P(\mathbf{x}_{k+1} = j | \mathbf{x}_k = i, \mathbf{u}_k)$ is the conditional probability that can be obtained from the motion model.

ii. *Update*: In this step, using Bayes' theorem, information in the new observation $\mathbf{z}_{k+1}$ is fused with the prediction to obtain the new belief at time $k+1$ as follows:

$$\begin{aligned} p_j(k+1) &\triangleq P(\mathbf{x}_{k+1} = j | \mathbf{z}_{k+1}) \\ &= \frac{P(\mathbf{z}_{k+1} | \mathbf{x}_{k+1} = j) \overline{p_j(k+1)}}{\sum_{i=1}^{M} P(\mathbf{z}_{k+1} | \mathbf{x}_{k+1} = i) \overline{p_i(k+1)}} \end{aligned} \tag{62}$$

where the conditional probability $P(\mathbf{z}_{k+1} | \mathbf{x}_{k+1} = i)$ can be obtained from the sensor model.

SHOUDONG HUANG
GAMINI DISSANAYAKE

University of Technology Sydney,
Sydney, NSW, Australia